1'p

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/020,286	12/11/2001	Alireza Dabagh	13768.213	8376

47973 7590 06/07/2006

WORKMAN NYDEGGER/MICROSOFT
1000 EAGLE GATE TOWER
60 EAST SOUTH TEMPLE
SALT LAKE CITY, UT 84111

EXAMINER	
CHUNG, JI YONG DAVID	
ART UNIT	PAPER NUMBER
2143	

DATE MAILED: 06/07/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	**Application No.** 10/020,286	**Applicant(s)** DABAGH ET AL.
	Examiner Ji-Yong D. Chung	**Art Unit** 2143

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *21 December 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-68* is/are pending in the application.

4a) Of the above claim(s) *38-65 and 68* is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-37, 66, and 67* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *12/6/2005*.	6) ☐ Other: _____.

9r c

DETAILED ACTION

General Comments

1. The Office has raised a number of potential issues with the instant application.

The main issue is with the use of various terms in the claims. For example, terms such as "notification", "verification," and "insertion of filter instance" can carry multiple meanings. A calling program can be "notified" of another program's successful termination, if the calling program invokes the second program that returns a value on a computing stack. A program maybe notified of another process by an event generator (e.g., semaphore).

Verification may correspond to a correct return value of a function (e.g., there is no error message upon a function invocation). Alternatively, a verification maybe performed by a program that explicitly performs error checks. A verification process may correspond to the compilation process of a filter graph into binaries, because during the compilation, there is a checking of correct graph structure. All of the preceding "verifies" the creation of a filter stack with the filter instance properly inserted.

Insertion of filter maybe performed in many ways. One may redefine a filter graph, using API that is described in Shaw et al. (Pat. No. 6,205,492) and allow the driver system to instantiate the description afforded by the graph. Insertion is performed at the graph level. At runtime, the whole filter stack is instantiated. Alternatively, one may instantiate a filter during runtime.

Amendments with additional language that further clarifies the types of issues presented above will help in advancing the prosecution.

Telephone Interview

2. The Office requests the applicant, at the applicant's convenience, for a telephone interview, to discuss the direction of the amendment. Contact information is provided at the end of the instant Office action.

In addition, the Office would like to discuss the publication date of NDIS 6.0.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

4. **Claims 1-15, 17-25, and 66** are rejected under 35 U.S.C. 102(a) as being anticipated by Machin et al (Pat. No. 6,243,753, Machin hereinafter), which incorporates by reference Shaw et al (Pat. No. 6,205,492, Shaw hereinafter).

With regard to **claim 1**, Machin shows, *in a computing device that may be communicatively coupled to a network by a communication path that includes one or more protocol stacks* [See lines 40-53 in column 2 for protocol stacks. Each protocol stack is a transport protocol driver. See Fig. 20 for a RDT] *associated with an abstract interface for managing a filter stack* [See item 422 in Fig. 20 and 162 in Fig. 6], *the filter stack including one more filter instances, which may perform filtering operations on data packets being transferred via the communication path* [See Fig. 21B, which shows two filter instances].

Machin does not mention pausing or starting the operation of the filter stack. Shaw does speak about the steps.

an act of pausing operation of the filter stack [This is because the new filter that is inserted in place, has all pins in the stop state. See Table 2 for the various states a pin can be in];

an act of starting operating of the filter stack [Once the interconnections have been specified and the filters instantiated (along with the interconnection stacks), they are in acquire state, about to run. See lines 12-42 in column 20.

However, Shaw indicates that that a filter stack maybe modified or edited (See lines 27-29 in column 23, after use. Shaw also mentions extensible system (See lines 19-25 in column 23), which is read as augmenting the instant filter system with additional driver ("inserting a filter object"). That is, Shaw shows *an act of inserting the filter instance into the filter stack.* In the context of Machin et al., Shaws's system inserts the filter *while at least one associated protocol stack continues to be capable of transferring data*, because Shaw states that the state transition need to be occur only on those which are in the stop state. See lines 38-42 in column 20. In other words, the any pause in processing is local to the filter stack.

Note that RDT and filters mentioned in Machin is the same system described in Shaw. See items 490, 498, and 496 in Fig. 20 of Machin.

With reference to **claim 2**, Shaw shows *the feature of an act of pausing one or more filter instances included in the filter stack.* Shaw does not mention this step, but the examination of his system reveals the feature.

As soon as one creates a new filter (in Shaw's system) and it is to be inserted, all the pin instances are in stop mode. Connecting any of the existing filter in the filter stack can occur in two ways: (1) the existing filter continues to operate or (2) it is placed in the stop state, to accommodate the changes in the filter stack design (assuming that it is not paused), will stop the existing filter from further processing, as the stack will be filled.

The second case need not be discussed, because that satisfies the limitation. If the Shaw's system attempts to connect the existing filter stack directly to a new extension, then, one or more filters in the stack will be placed in the paused state, because the buffer associated with the new filter will quickly fill (Remember, it is not in acquire more). Any producer process associated with other filters will pause.

With reference to **claim 3**, Shaw shows that pausing one more or more filter istances included in the filter stack comprises the following:

an act of one or more pause routines receiving data indicating the one or more filter instances should be paused [See lines 47-57 in column 14, Table 2 and lines 48 in column 18 to lines 10 in column 19. 'Set' function type is are the ones that set specific property ("data") regarding the pin states, including stop, to acquire and run properties]. The 'set' routine receives property ("data") that indicates the state (which includes stop, run, pause, etc) as inputs.

With reference to **claim 4**, Shaw shows that the act of pausing operation of a filter stack comprises the following: an act of redirecting a transferred data packet to a dummy routine that returns the data packets to the communication path without modifying the data included in the data packet

Creating a filter and placing the filter with its pins in stop state means that the data has been passed (redirected) to a stack associated with the output pin of a preceding filter. The stack (not the same thing as the filter stack) is the dummy routine; it returns the data packet to the communication path without modifying the data included in the packet.

With respect to **claim 5**, Machin shows *the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following*:

an act of inserting a filter instance that was configured by using parameters received from the abstract interface [In accordance with the above explanation of "inserting a filter instance" related to claim 1, "abstract interface" is item 492 in Fig. 20. The recreation of a filter stack, necessary for the instantiation of the insertion, requires the reception of a virtual connection ID from the 'integrating component' (which is part of the "abstract interface.") See lines 38-53 in column 28. That is, the filter is configured with parameter received from the interface].

With respect to **claim 6**, Shaw shows the method as recited in claim 1, *wherein inserting the filter instance into the filter stack comprises the following*;

an act of a filter driver receiving a filter handle that may be used to facilitate transferring data to an abstract interface [See lines 56-67 in column 10. It shows that the filter handle is created from the driver object ("filter driver") by invocation of create handler, which in turn, results in the invocation of create handle. That is, the filter driver "receives" the filter handle through the invocation of create handle. See Fig. 21A and 21B. for the flow of data from the interface].

With respect to **claim 7**, Shaw shows the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:

an act of a filter driver allocating resources for the filter instance. See lines 1-11 in column 20. The instantiation of a filter (for the filter stack) and interconnection requires the creation of stacks. Each stack is a "resource."

With respect to **claim 8**, Shaw shows *inserting the filter instance into the filters stack comprises the following:*

an act of a filter driver creating a filter instance context for the filter instance. See lines 20-35 in column 9. It states, "A file object with appropriate context information will be created." Note that a file object is the instance of the filter.

With reference to **claim 9**, Machin shows *the method as recited in claim 8, wherein a filter driver creating a filter instance context for the filter instance comprises the following:*

an act of the filter driver sending the filter instance context to the abstract interface.

However, Machin also shows that that the integrating component ("the abstract interface:") is NDIS, in which components (including filter drivers) register their interfaces. See lines 36-51 in column 4. The registration process in NDIS sends in all information that is required for making calls (i.e., context).

With reference to **claim 10**, Machin shows the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:

an act of a filter driver registering data with the abstract interface.

Machin shows NDIS, which is used to register filter driver. See lines 36-51 in column 4.

With reference to **claim 11**, Machin shows the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:

an act of a filter driver registering data in a system registry.

In Windows 95 or NT, as mentioned in Machin, filter driver information is registered to the system registry. This occurs either directly or through NDIS. They maybe accessed by NDIS.

With respect to **claim 12**, Shaw and Machin shows the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:

an act of inserting a filter instance that was configured by using parameters received from a system registry.

In Windows, as indicated earlier, information about filter drivers is stored in the system registry. Instantiation of a filter in Windows operating systems uses the filter (driver) information in the registry.

Wit respect to **claim 13**, Shaw and Machin show *that inserting the filter instance into the filter stack comprises the following:*

an act of inserting the filter instance in a predetermined location in the filter stack.

As it has been discussed before, Machin and Shaw show the insertion of filter instance into a "stack." The location of the insertion cannot be otherwise than in accordance with a "predetermined" location designated through the user API mentioned in Shaw.

With reference to **claim 14**, Machin shows the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:

an act of inserting a filter instance that is capable of filtering data packets transferred over virtual connections.

See lines 32-49 in column 27 of Machin for the mention of virtual connection.

With reference to **claim 15**, Machin shows the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:

an act of inserting a filter instance that includes ala entry point to receive data associated with the power management of the computing device.

As indicated with respect to other claims, Machin mentions Windows NT and NDIS.

In Windows operating system (e.g., Windows 2000 published before the filing of the instant application), power management delivers I/O information to device drivers ("filter drivers"). With NDIS, an entry point exists for every filter.

With reference to **claim 17**, Shaw shows *the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:*

an act of inserting a filter instance that includes properties that may be modified through a management interface. See in Shaw, from line 12 in column 13 to line 60 in column 14. See also Table 2.

With reference to **claim 18**, Machin and Shaw show the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:

an act of inserting a filter instance on a filter stack that is bound to a plurality of transport layer protocols.

See lines 49-52 in column 28, which states that the same filter instance may receive or send data to different connections. Such reception and transmission to different connections means that there are different bindings to different connections ("bound to a plurality of transport layer protocols.")

With respect to **claim 19**, Machin shows the method as recited in claim 1, wherein inserting the filter instance into the filter stack comprises the following:

an act of a filter driver verifying that the filter instance was inserted into the filter stack.

With NDIS, all filter instances transmit their status changes to the NDIS, which in turns "verifies" to the protocol that it has been instantiated. The invocation of NDIS's status change interface in turns calls protocol status function (e.g., ProtocolStatus). This lets the protocol driver know that it has been reset or that its status has been changed. Thus, every filter driver in NDIS verifies the presence of the filter instance.

With respect to **claim 20**, Machin shows the method as recited in claim 19, wherein a filter driver verifying that the filter instance was inserted into the filter stack comprises the following:

an act of the filter driver sending an insertion status to the abstract interface.

As explained with reference to claim 19, the filter driver's notification to the protocol stack is through NDIS ("abstract interface.")

With respect to **claim 21**, Shaw shows *the method as recited in claim 1, wherein starting operation of the filter stack comprises the following:*

an act of starting one or more filter instances included in the filter stack.

See lines 12-31 in column 20 of Shaw. The filer instance changes their status from stop to acquire state.

With respect to **claim 22**, Machin shows the method as recited in claim 21, wherein starting one or more filter instances included in the filter stack comprises the following:

an act of one or more start routines receiving data indicating that the one or more filter instances should be started.

The starting of the filter instances in NDIS is performed by initialization. After the initialization, packets can be sent to it via NDIS interface. The initialization is a routine, and it receives data indicating the filter stack should be started.

With respect to **claim 23**, Machin shows *the method as recited in claim 1, further comprising:*

an act of notifying associated protocol stacks that operation of the filter stack is going to be paused.

Machin mentions NDIS. NDIS provides various interfaces for propagating, from the driver, that the calling filter driver will be paused.

With respect to **claim 24**, Machin shows *a method as recited in claim l, further comprising:*

an act of notifying associated protocol stacks that the filter stack now includes the inserted filter instance.

The instantiated filter's presence is notified to the protocol stack by the invocation of the NDIS interface to open a particular filter (i.e., 'adapter'), to which NDIS can return with success or failure "notification." The success indicates that the "filter stack" includes the filter instance.

With respect to **claim 25**, Machin shows the method as recited in claim 1, further comprising

an act of notifying associated protocol stacks that operation of the filter stack is going to be started.

NDIS provides means to notify the start of the filter stack, by a return value upon the invocation of the initialization routine against the filter driver. The success return value indicates that it is going to be started.

Claim 66 substantively incorporates the limitations of claim 26, but claims subject matter as computer software. The reasons for the rejection of claim 1 apply to claim 66.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. **Claim 16** is rejected under 35 U.S.C. 103(a) as being unpatentable over Machin and Shaw in view of Shilmover et al. ("Windows 2000 Power Toolkit", Shilmover hereinafter).

With respect to **claim 16**, Machin and Shaw show the method of inserting a filter instance, but does not indicate that the instance includes means for receiving data from plug and play.

Shilmover uses filter driver for *plug-and-play* and NDIS.

It would have been obvious to one of ordinary skill in the art at the time of the invention to implement filter stacks that permit the insertion of filter instances for processing plug-and-play data, in Windows 2000, because Machin and Shaw shows that (1) the filter stack is adapted for Windows NT and (2) Windows 2000 supports filter drivers for PNP.

The motivation for the combination is to extend the advantages of filtering technology, as explained in Shaw's summary and background, is extensibility interoperability. See lines 20-43 in column 4 of Shaw.

7. **Claims 26-37 and 67** are rejected under 35 U.S.C. 103(a) as being unpatentable over Machin in view of Miloushev et al (Pub. No. US 2003/0056205, Miloushev hereinafter).

With reference to **claim 26**, neither Machin nor Shaw mentions that a filter maybe removed. However, the above claims have spoken about extending (i.e., adding filter instance) a filter stack by using the API that Shaw mentions.

Miloushev shows filter diagram editing system, in which filters (or components) are *added or removed* to a design.

It would have been obvious to one of ordinary skill in the art at the time of the invention to augment Machin and Shaw's system, such that filters maybe removed (as well as added through user API, which Shaw shows), because being able to remove a filter allows one to make corrections to correct filter stack designs.

Claims 27-29 substantively incorporate the limitations of claims 2-4. The reasons for the rejection of claims 2-4 apply to claims 27-29.

With respect to **claim 30**, Machin and Shaw show the act of pausing the filter instance. In Machin and Shaw's system, pausing the filter stack pauses all filter instances within.

With respect to **claim 31**, Machin and Shaw show that the act of removing the filter instance comprises an act of a filter driver releasing resources associated with the filter instance. As indicated above, Machin and Shaw mention NDIS as the abstract interface. The interface requires the filter drivers to support a shutdown operation, which releases system resources.

Claims 32-36 substantively incorporate the limitations of claims 21-26. The reasons for the rejection of claims 21-26 apply to claims 32-36.

Claim 37 differs from claim 26 in that claim 37 defines reconfiguring filter stack in the manner that promotes efficient transfer of data. Claim 26 shows a removal of a filter, which promotes more efficient transfer of data locally, because the removal of the filter reduces the amount of computation. Note that an overall efficiency is not considered for the rejection of the claim.

Claim 67 substantively incorporates the limitations of claim 26 but claims the subject matter as a software product. The reasons for the rejection of claim 26 apply to claim 67.

Conclusion

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ji-Yong D. Chung whose telephone number is (571) 272-7988. The examiner can normally be reached on Monday-Friday 9:30-6:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, David Wiley can be reached on (571) 272-3923. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Ji-Yong D. Chung
Patent Examiner
Art Unit: 2143

[illegible]
DAVID WILEY
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

JDC